On Developing Integrated Systems Architecture and Systems Engineering Courses at RIT

Wayne W. Walter, Paul H. Stiebitz
Rochester Institute of Technology

Introduction

The Rochester Institute of Technology (RIT), the Massachusetts Institute of Technology, the University of Detroit Mercy and the Naval Post Graduate School have joined with industry leaders to create a two-year product development program at the Master's Degree level for mid-career technical managers. Using a common curriculum framework, each member of the consortium, named the Educational Consortium for Product Development Leadership in the 21st Century (PD21), customizes course materials and elective courses to meet the needs of their respective constituency. The program balances technical and business perspectives in an effort to provide technical leaders with the skills and knowledge to create best-in-class product portfolios.

The program at RIT, known as the Masters in Product Development (MPD), is a joint effort between the College of Business and the Kate Gleason College of Engineering. In addition to a course in Leadership in Product Development, the core of the curriculum consists of three systems design and management courses: Systems Engineering (SE), Systems Architecture (SA), and Systems and Project Management. Students are also required to complete courses in Systems Optimization, Engineering Risk-Benefit Analysis, Operations Management, Organizational Processes, Marketing Management and Finance and Managerial Accounting, in addition to four elective courses. The program spans two calendar years beginning with an intensive five day per week, month long session in January that focuses on fundamental concepts in product leadership and organizational behavior. Students subsequently take two courses per quarter in an "executive" format consisting of one course in the morning and one in the afternoon, one day each week. The student cohort formed in the January session takes all courses together, except possibly for two electives at the end of the program.

Students accepted into the MPD program must have at least five years of product development experience, although most have between 10 and 20 years of experience. Many already have earned advanced degrees. The maturity of the students and the "executive" format of the MPD program creates challenges in both course design and delivery. For example, topics must be aggregated and partitioned so that discussions end cleanly, student assignments must be carefully defined and synchronized with the materials covered each week, and an assortment of delivery methods must be employed in order to effectively engage students over a four hour period.

The sequencing of courses in the program, creates additional challenges for the Systems Engineering and Systems Architecture courses. The development of a complex system logically begins with market and customer requirements, strategy and scoping, concept generation, technology integration, function partitioning and component aggregation. These are aspects of systems architecting, yet the expression of the architecture depends on systems engineering tools and methods. Currently, the SE course precedes the SA course, a sequence that seems counterintuitive to some.

This paper discusses the collaborative development of the Systems Engineering and Systems Architecture courses at RIT. It focuses on the co-development of course objectives, the coordination of topics, the use of case studies and a joint team project.

Overview of the Systems Engineering Course Objectives, Content and Delivery

The top-level objective of the Systems Engineering course is to provide students with a firm understanding of the critical role played by systems engineers in organizations that develop complex products. Educating for system thinking and problem solving is the primary objective to insure that key leaders are comfortable and highly functional when dealing with complexity. Systems Engineering is a systematic treatment of the systems engineering process, providing methods and tools to be used in the Systems Architecture course, and the Systems and Project Management course, which are taught in subsequent quarters.

Topics covered include System Engineering Basics, Requirements Analysis, Trade Studies, Case Studies, Software Engineering, Design-to-Cost, Modeling and Simulation, and Robust Design. A term-long design project is employed to improve the depth of understanding of all aspects of the SE course, and this will be covered in detail in a later section of this paper.

In Systems Engineering Basics the learning objectives include having students understand the benefits of System Thinking, how SE deals with complexity, SE concepts and language, the need for SE standards, and various SE process models. Emergent properties, complexity of detail and dynamic complexity, feedback, time delays, and side effects are covered, as well as examples of well-known standards and SE process models.

In order to understand the importance of Requirements Development and Management, user and systems requirements are discussed in detail in addition to business requirements and QFD. Requirements flow-down, traceability and validation are covered. The translation of requirements into functional and physical architecture is discussed using various diagrams including Functional Flow Block Diagrams (FFBD), Functional Analysis and System Technique (FAST) Diagrams, Timing Diagrams, and State Transition Diagrams.

The importance of Trade Studies is emphasized. Methods and tools covered include the Pugh Technique for concept selection, Failure Modes and Effects Analysis (FMEA), Fault Tree Analysis (FTA), and TRIZ. A discussion or demonstration of process modeling and simulation is included to show it's importance as a decision making tool.

In order to further improve the student's depth of understanding of the SE concepts discussed above, case studies on a printer/copier, and a personal digital assistant are discussed with the class. Student teams are given a comprehensive list of SE tools and subprocesses, and are asked to select the tools and subprocesses and their sequence leading to a recommended solution to the problems given. A basic understanding of Design-to-Cost is also achieved through a discussion of a case study.

Approximately half of our students have a software background, so a substantial emphasis is placed on software engineering issues. The learning objectives here are to raise the awareness of the processes, methods, and tools for complex software systems development, and to gain an understanding of Object Oriented (OO) Technology and the Capability Maturity Model (CCM).

A basic treatment of Robust design is included in order to introduce students to structured experimentation using Orthogonal Array Methodology, the power of parametric design, and tolerance design.

The scope of the course is rather broad, so that one book is not sufficient to cover all of the above topics in sufficient depth. As a result, a multitude of well-known texts is used including those by Blanchard and Fabrycky[1], Martin[2], Hatley and Pirhbai[3], the INCOSE SE Handbook[4], and Stevens et al[5]. It is taught in a team format with faculty from the departments of Mechanical and Software Engineering, and the Center of Quality and Applied Statistics, and is being developed with the input from a systems engineer with many years of top-level systems experience engineering complex products for one of our industrial partners.

The course is structured around a generic systems engineering process, although the process used at the students' company is mapped onto the generic process, so students understand where their process fits within the discussion. For a term project, student design teams are asked to develop a feasible concept for a product using the generic process. As various portions of the generic process are discussed in class by way of a class example, students are asked to apply these sub-processes on their project. In addition to the project, students are assigned individual and group problem sets to practice the concepts presented.

Class lectures have been developed in PowerPoint™ software for ease of projection in class, although students are provided with a loose-leaf binder with hard copies. Each lecture includes time for class discussion, since our students have considerable industrial experience and bring much expertise to a class discussion. In addition to project presentations at the conclusion of the course, some opportunities for student class presentations are made available to promote student interaction. Guest speakers are scheduled to complement class material.

The intent of the systems engineering course is to provide students with the mindset, logic, and tools needed by systems engineers and architects in organizations that develop complex products. Systems Engineering has been developed using the MIT model, but it has been tailored

to the consumer product needs of our customers here in Rochester. Much time and effort has been devoted to developing a top-quality effective presentation to delight our customers.

Overview of the Systems Architecture Course Objectives, Content and Delivery

The Systems Architecture course is a high-level, systemic treatment of the design of systems, with emphasis on product families and product platforms. It endeavors to improve the architecting skills of technical experts who lead the creation of superior systems.

The top-level goals of the course are partitioned into six categories: Systems Thinking, Principles of Systems Architecture and Architecting, Product Architectures and Organizational Processes, Architecting for "X", Software Architectures, and Digital System Architectures.

"Holistic," "end-to-end," and "top-sight" are some of the terms that motivate the systems thinking component. Systems science concepts, such as coupling, cohesion, feedback, unintended effects, transience, and adaptive behavior are discussed in the context of product and process design choices.

To achieve superior product development results, architecture and architecting processes must be consistent with the business strategy, organization, and organizational processes. Consequently, the generic steps of architectural development are addressed in the context of the business environment. Emphasis is placed on product family strategies, architectural structures (such as the choice of modular or integral designs), evaluation of architectures, and the relationship between the product structure and the enterprise structure.

Architecting for "X" (AFX) is the concept that requirements pertaining to international markets, manufacturability, reliability, serviceability, usability, environmental performance, etc., must first be addressed in the architecting phase. The need, opportunity, and process of incorporating these attributes into the architecture, are discussed.

The ubiquity of digital systems motivates the discussion of how software and computer architectures impact product performance, consumer perception, and product development. Software architecture concepts, such as heuristics, quality attributes, patterns, styles, and reference models are introduced. Digital system architectures, such as application specific IC's, programmable logic devices, microprocessors, microcontrollers and multi-processors, are surveyed.

The course is team-taught by faculty from the Industrial and Manufacturing Engineering, Electrical Engineering, and Software Engineering Departments. Weekly class sessions are a blend of lectures supported by PowerPoint™ slides and handouts, discussions based on articles chosen by students, and case discussion. While case discussion is commonplace in business education, it tends to be a missing or underutilized technique in most engineering curricula. We have found case discussion to be a very effective way to engage the substantial experience of students while forestalling the onset of tedium associated with long class sessions. Finally, a

team project, discussed below, provides an opportunity for students to apply course concepts to a realistic product development scenario.

Because there is no text well suited to the course, one is currently under development. In the meantime, readings have been assigned from three texts: *The Art of Systems Architecting*[6], and *Software Architectures in Practice*[7], and *Structured Computer Organization*[8], along with numerous journal articles.

Joint Team-based Project and the Connection with Course Objectives

In order to facilitate a more in-depth understanding of the concepts, and to practice the methods and tools discussed in the two courses, a project was assigned in the SE course that was continued in the SA course. "Robodog", a robotic guide dog for the visually impaired, was selected because it involved extensive software, as well as hardware, issues. Only about 2% of blind individuals have living guide dogs for a variety of reasons. Many are elderly and feel they cannot take proper care of a dog, while others feel that they are not truly independent if they rely on a dog to get around. For these reasons, and because of the advanced state of current technology, SE students were asked to work in teams of four, and use a generic SE process model to develop a feasible concept for a navigation and collision avoidance device to allow a blind individual to get around and lead an independent lifestyle, yet be an alternative for a living, breathing guide dog.

With much of the preliminary work completed for Robodog at the end of the SE course, SA students were asked to go back and consider the platform issues appropriate to developing a series of product releases with various features of increasing sophistication. While these architecting tasks would, in practice, be completed before undertaking the more detailed systems engineering tasks, these details developed in the SE course provided students with a rich understanding of the architectural issues.

Some of the specific deliverables required in the two courses, will now be covered in some detail.

SE Course

Efforts in the SE course were initiated by a visit from Karen Vetters, a 40-year-old blind female, her guide dog Shelby, and a representative from Upstate Guide Dog Association, an organization which raises and trains dogs for customers in Hilton, NY. Karen explained in some detail what her dog does for her, and why she thought that an electro-mechanical device would never provide the companionship that Shelby provides to her. Her customer perspectives were invaluable to the students, and provided a better understanding of customer requirements for such a device.

SE students were then asked to do some research on the Internet and by phone to discover additional user requirements that they felt their product should meet, in addition to those in the Initial Requirements Document that was provided to the students at the beginning of the class.

From this, they were asked to create a base-line Requirements Document. This was part of the Requirements Analysis portion of the generic SE process model. At the same time, teams were asked to complete an Environment and Use Scoping Phase that defined the interactions Robodog was to have with the environment (i.e. elements outside the system they were developing). By defining the system boundary, students were able to decide what was inside and what was outside their system. As part of this effort, User Scenarios were developed to depict interactions between the users and the system from the standpoint of "what a user is trying to accomplish," and Control Context Diagrams (CCD) and Data Context Diagrams (DCD) were constructed.

In the Function Scoping Phase, which followed, students determined the system functional requirements from the user requirements by developing a QFD House 1, and they developed a FAST Diagram that captured the system level critical parameters. This was part of the Concept Analysis Portion of the generic SE process model.

The Function Scoping Phase was followed by the Requirements Documentation Phase in which teams were asked to formalize the user, systems, and mission requirements by documenting them in the User Requirements Document (URD), the Systems Requirements Document (SRD), and the Operational Requirements Document (ORD). They were also asked to develop a recommendation for a Requirements Management and Traceability mechanism that would document the source of various requirements, allow them to be traced down into various system levels after "flowdown," and allow Requirements Change Management.

In the Concept Generation and Selection Phase, a trade study, using the Pugh Technique, comparing the attributes of the alternative concepts was done to identify the "best" concept, as part of the Functional Analysis portion of the generic SE process model. An "artist's rendering" of the selected concept was done at this point, and high level specifications such as weight, size, ground speed, mobility, pull force, cost, and reliability were developed. A function and behavior analysis to capture the sequencing of functions necessary to satisfy customer requirements was done by creating a Functional Flow Block Diagram.

Software-Based Diagrams including an Object Diagram, Sequence Chart, Activity Diagram, and Use Cases at the system level were done for the entire system including the software. The integration of the hardware and software was worthy of a close look by the teams at this point.

At the final design review, teams presented a 25 min overview of the their product top-level architecture, including a certification analysis and a verification and validation discussion. The certification analysis examined how the Robodog architecture might fail to deliver the performance specifications. A FMEA or FTA analysis was chosen. The purpose of the verification and validation phase was to insure the design met system and mission requirements, and to insure the design was what the customer wanted. This constituted the large feedback loop back to Requirements Analysis on the generic SE process model.

Overall student reaction to the project was good. Students struggled initially to understand what the deliverables were, and they complained that it required a lot of work, but most agreed that it was necessary to practice and internalize the methods and tools presented in the lectures.

SA Course

In the Systems Architecture course, the students were given the tasks of developing a product family strategy and attendant top-level architecture for Robodog. As such, they were asked to complete five major deliverables: a market assessment, technology and platform roadmaps, a modularity definition, an architectural definition and an architectural review. The objective of the market assessment task was to define a market segmentation model that would serve as a basis for a product platform and family plan. Students were asked to extend their research regarding both existing and potential markets for Robodog, and then develop a market segmentation model consisting of 2-4 price/performance tiers and 2-4 market cohorts. They were then asked to consider current and obvious application opportunities for Robodog, as well as future applications. For each segment, they developed high and low estimates of the total market yearly sales volume. They identified leading technological and product competitors along with the most critical customer needs in each market segment.

The objective of the technology and platform roadmap task was to create a vision of how the technological advances could impact future generations of Robodog. Students developed a 10-year technology roadmap representing three to five key technologies covering three to five Robodog generations. These were then mapped onto the market segmentation model.

The objective of the module definition task was to create function partitions that enable the execution of the technology and product roadmaps. Students factored in thirteen different modularity drivers that are based on technical, organizational and marketing considerations.. For each module, students identified the functions performed, the interfaces with other modules, and the modules common across the product family.

The objective of the architecture definition task was to define key aspects of the top-level architecture of Robodog in a way that is consistent with the market assessment, the technology platform plan, the product platform plan and the module definition. Students constructed Architectural Flow Diagrams (AFD) to show the information flows among modules and the Architectural Block Diagram (ABD) to depict the subsystem structure of the product family architecture.

The objectives of the architectural review were to disclose the product platform strategy and to assist in the architectural certification. Certification is the process by which product developers gain confidence that customer requirements are being met. In a 15-minute presentation, students reviewed their market segmentation model, technology and product platform roadmaps, the architectural flow diagram, the architectural block diagram, and they were asked to demonstrate that the critical customer requirements were captured in the product family concept.

Cooperation With Other Institutions

The MPD program has been influenced in varying degrees through RIT's association with the Educational Consortium for Product Development Leadership in the 21st Century (PD21), and with the Center for Innovation in Product Development (CIPD). PD21 currently consists of RIT, the Massachusetts Institute of Technology, The University of Detroit Mercy and the Naval Post Graduate School. CIPD is an industrial consortium currently consisting of CVC, Inc., Ford Motor Company, General Motors Corporation, IBM Corporation, Ide, Inc., ITT Industries, The National Science Foundation, Polaroid, Product Genesis, Inc., U.S. Navy, and Xerox Corporation.

The SE and SA curriculum was most heavily influence by Xerox, a founding partner of CIPD and sponsor of 20 of 21 students in the first MPD class. Prior to the beginning of the program, comments on the SE and SA curricula were solicited from Xerox's Chief Engineer and his staff. Numerous recommendations were incorporated into the SE course with the help of a recently retired senior systems engineer and guest speakers. Other recommendations produced a fundamental shift in the SA course away from engineering analysis and toward the strategic issues related to product platform and product family definition. Guest speakers from Xerox and Lockheed Martin provided supplemental lectures in product innovation, QFD and Design-to-Cost.

Cooperation with other universities has been more challenging. Two PD21 workshops were held at MIT to discuss the development of their program. Additionally, the MIT professor responsible for the SE course visited RIT to discuss the role of systems engineering in product development and provided copies of classroom materials for review purposes. Although RIT was not permitted to use these class materials, this assistance was very helpful in defining the scope and content of our own SE course. The development of the SA course departed significantly from the MIT version due to differences in industrial populations served by the two universities. MIT's program draws heavily from the aerospace and government sectors while RIT draws from the commercial and consumer product sectors.

Both PD21 and the CIPD consortiums continue to meet periodically to share program accomplishments and plans for the future.

Conclusions

The International Council on Systems Engineering (INCOSE)[9] defines systems engineering in the following way:

"Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem: Operations, Performance, Test, Manufacturing, Cost & Schedule, Training & Support, and Disposal. Systems Engineering integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept to production to operation. Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs"

From this definition, it is apparent that courses entitled “Systems Engineering” and “Systems Architecture” are just different facets of a broad systems engineering field. For this reason, and because systemic thinking is a core aspect in the program, it is essential that these courses be carefully coordinated. The following are some additional challenges encountered throughout the past two years are listed as follows:

- Students have experiences they want to share, and time needs to be made available to accommodate their input and discussion. This can mean a re-shuffle of lecture slides “on-the-fly” as the discussions cover some lecture slides, and others need to be cut when time runs out.
- Students are continually challenging and testing the lecture material to see its relevance to them as individuals in the organization in which they are employed.
- Students have high grade expectations. Many are managers with multiple individuals reporting to them. Some are systems engineers and architects with many years of experience. Many expect that A is the norm grade for them.
- Students are under pressure from job responsibilities they must continue to cover, as well as family demands on their time. They need to be convinced that assignments are well worth their time to complete, and “add value.”
- Students are reluctant to downgrade a team member, so group and team project grading must be balanced with individual problem assignments.
- Project topics need to interesting enough to keep students interested through two quarters.
- Case studies are well received by students, but difficult to obtain, as companies are reluctant to release information that may not be entirely complementary. Many of our case studies need to be generic and constructed to illustrate the important issues, methods, and tools, discussed. This is time intensive. In the SE course, we have been fortunate to have help here from a top-level systems engineer and project manager who recently retired form one of our industrial partners, who is creating our case studies and leading class discussions about them.
- Students find a multitude of texts to be intimidating, and would prefer a single text which does not exist.
- Students come from a variety of companies and backgrounds. Some are engineers with primarily hardware experience. Some are computer scientists and software engineers with extensive software backgrounds. Others are chemists, with little engineering experience. It is a challenge to engage all students in the material, as some are seeing it for the first time, and others have extensive experience with it.

In order to deal with these challenges, we have established an ad hoc committee that includes the MPD program director and the faculty member responsible for the Systems Optimization course. This committee meets regularly to discuss the top-level objectives of the systems core, to compare and contrast topics included in the curricula, to find synergies among course materials and to identify gaps. We have utilized a structured method, drawn from Magher[10], to translate top-level objectives consistently into content and student assignments. We have attended each other’s class sessions, and we have created a joint team-based project. These efforts have produced significant benefits in the selection, partitioning and coordination of topics, and a group approach to dealing with issues above, as well as new issues as they arise.

Bibliography
[1]Blanchard, B.S. and Fabrycky W.J. *Systems Engineering and Analysis.* Upper Saddle River, NJ: Prentice Hall (1997).
[2]Martin, J.N. *Systems Engineering Guidebook*. New York, NY: CRC Press (1997).
[3]Hatley, D.J. and Pirbhai I.A. *Strategies for Real-time System Specification.* New York, NY: Dorset House Publishing (1987).
[4]International Council on Systems Engineering. *Systems Engineering Handbook* (1999).
[5]Stevens, R., P. Brook, K. J., and Arnold, S. *Systems Engineering: Coping with Complexity.* London, UK: Prentice Hall Europe (1998).
[6]Rechtin, E., and Maier M. W. *The Art of Systems Architecting*, CRC Press, FL (1997).
[7]Bass, L., Clemens P. and Kazman C. *Software Architectures in Practice,* Reading, MA: Addison-Wesley (1998).
[8]Tannenbaum, A.S. *Structured Computer Organization*. Upper Saddle River, N.J.: Prentice Hall (1999).
[9]URL: http://www.incose.org/whatis.html.
[10]Mager, R.F. *Making Instruction Work.* Belmong, CA: Lake Publishing Company (1988).

PAUL H. STIEBITZ
Paul Stiebitz is an associate professor in the Industrial and Manufacturing Engineering Department at the RIT where he teaches courses in product development, simulation and optimization. Prior to joining RIT in 1984, Paul developed optical and electrophotographic systems for copiers for 15 years at Xerox and Kodak.

WAYNE W. WALTER
Wayne Walter is the James E. Gleason Professor of Mechanical Engineering at RIT. He received his Ph.D. in Mechanics from Rensselaer Polytechnic Institute. Wayne has worked for the U.S. Army, Rochester Products and Delco Products Divisions of General Motors, and Xerox, and is a registered professional engineer (P.E.) in New York State.